Package 'SCAtools'

September 18, 2026

Type Package

Title Direction-Aware Sufficiency Condition Analysis

Version 0.4.3

Date 2026-09-10

Description Provides a direction-aware interface for analysing bivariate sufficiency statements from empty-space frontier patterns. Logical sufficiency directions (high or low levels of a condition and outcome) are kept separate from the physical location of the empty corner in the scatter plot. Computation is delegated to version 5 of the 'NCA' package based on Dul (2016) <doi:10.1177/1094428115584005>, using the contraposition between necessity and sufficiency. Threshold tables are computed in actual units and converted by this package, so percentage, percentile and standard-deviation scales follow one stated reporting convention in every sufficiency direction. Includes tidy summaries, threshold rules, plots, random-data generation, permutation tests, and power analysis. An ordinary least-squares line can be drawn beside the frontier as a central-tendency reference; it is an average-effect summary and never a component of a sufficiency claim. An empty-space pattern alone does not establish causality or deterministic sufficiency.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 3.5.0)

Imports ggplot2 (>= 3.4.0), NCA (>= 5.0.2), stats, utils

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/youngchanresearcher/SCAtools

BugReports https://github.com/youngchanresearcher/SCAtools/issues

NeedsCompilation no

Author Young Chan [aut, cre]

Maintainer Young Chan <youngchanresearcher@gmail.com>

Repository CRAN

Date/Publication 2026-09-18 11:40:11 UTC

Contents

`SCAtools-package` *Direction-Aware Sufficiency Condition Analysis*

Description

Tools for expressing bivariate sufficiency statements as empty-space frontier analyses while keeping logical directions separate from physical scatter-plot corners. Computation is delegated to **NCA** through contraposition.

Details

The four directions are `HH`, `LH`, `HL`, and `LL`; the first letter refers to X and the second to Y. Their physical empty corners are 4, 3, 2, and 1, respectively. Empty-space evidence alone does not establish causal or deterministic sufficiency.

See Also

`sca_analysis`, `sca_corner_map`

`sca_analysis` *Sufficiency Condition Analysis*

Description

Maps each logical sufficiency direction to the corresponding physical empty corner and calls the NCA frontier engine. `sca()` is a compact convenience wrapper.

Usage

```
sca_analysis(
  data, x, y, direction = "HH",
  ceilings = c("ce_fdh", "cr_fdh"), reference = NULL, custom = NULL,
  scope = NULL, threshold.x = "percentage.range",
  threshold.y = "percentage.range",
  convention = c("absolute", "directional"),
  steps = 10, step.size = NULL,
  cutoff = 0, qr.tau = 0.95, test.rep = 0,
  test.p_confidence = 0.95, test.p_threshold = 0.05,
  bottleneck.x = NULL, bottleneck.y = NULL
)

sca(
  data, x, y, direction = "HH",
  ceilings = c("ce_fdh", "cr_fdh"), reference = NULL
)
```

Arguments

`data`	A data frame or object coercible to a data frame.
`x`	One or more condition columns selected by name or position.
`y`	One outcome column selected by name or position.
`direction`	One direction or one per condition: `"HH"`, `"LH"`, `"HL"`, or `"LL"`.
`ceilings`	Empty-space frontier techniques accepted by `nca_analysis`, such as `"ce_fdh"` and `"cr_fdh"`. `"ols"` is not one of them: it is moved to `reference` with a warning.
`reference`	Central-tendency lines drawn beside the frontier for comparison, or `NULL` for none. `"ols"` is the ordinary least-squares regression of the outcome on the condition, fitted to all the data. It estimates no empty space, so it carries no effect size, no permutation test and no threshold table. See `sca_reference`.
`custom`	Optional intercept-slope pairs for custom frontiers.
`scope`	Optional theoretical scope `c(xmin, xmax, ymin, ymax)`.
`threshold.x, threshold.y`	Reporting scales for the condition and outcome axes: `"actual"`, `"percentage.range"`, `"percentage.max"`, `"percentile"`, or `"sd"`. See `sca_scales`.
`convention`	`"absolute"` places 0 at the low end of every axis and lets the inequality carry the direction. `"directional"` places 0 at the least sufficient end, mirroring low-level directions so that every rule reads as "more of the sufficient thing".
`steps`	Number of threshold steps, or an explicit vector of outcome levels expressed on the `threshold.y` scale.
`step.size`	Optional threshold step size on the `threshold.y` scale.
`cutoff`	Representation of out-of-range threshold values.
`qr.tau`	Quantile for the quantile-regression frontier.

`test.rep` Number of permutation resamples; zero skips testing.

`test.p_confidence`
Confidence level for p-value accuracy.

`test.p_threshold`
Significance threshold shown in test output.

`bottleneck.x`, `bottleneck.y`
Deprecated former names of `threshold.x` and `threshold.y`. A bottleneck is a necessity concept and does not describe a sufficiency statement.

Details

Directions in one call must share the same Y letter. Run separate models when both high-Y and low-Y statements are required. Effect sizes quantify fitted empty-zone area relative to the declared scope; they are not by themselves proof of causality.

Threshold values are always requested from the engine in actual units, and every scale conversion is performed by this package. The engine applies its own conventions inconsistently across scales and mirrors some of them according to the physical empty corner, which does not correspond to the logical sufficiency direction. Keeping the conversion here means one stated convention governs both axes in all four directions, and it lets a fitted model be re-expressed on any scale by `sca_thresholds` without refitting.

Value

An object of class `sca_result`.

See Also

`sca_table`, `sca_scales`, `sca_plot`

Examples

```
dat <- data.frame(X = seq(0, 1, length.out = 12),
                  Y = seq(0, 1, length.out = 12))
fit <- sca_analysis(dat, "X", "Y", direction = "HH",
                    ceilings = "ce_fdh")
fit

# Report the condition axis as percentiles instead
sca_thresholds(fit, scale = "percentile")
```

`sca_corner` *Map Sufficiency Directions and Physical Empty Corners*

Description

Converts between logical SCA directions and physical empty corners, returns the complete mapping table, lists the vocabulary of condition analysis in degree with the names this package uses for it, or lists the names retired in 0.4.0.

Usage

```
sca_corner(direction = c("HH", "LH", "HL", "LL"))

sca_direction(corner = 1:4)

sca_corner_map()

sca_terms()

sca_legacy_names()
```

Arguments

direction	Character vector containing "HH", "LH", "HL", or "LL". The first letter refers to X and the second to Y.
corner	Integer vector of physical corner numbers from 1 through 4: upper-left, upper-right, lower-left, and lower-right.

Details

sca_corner_map() reproduces the sufficiency half of Table 1 of the condition analysis in degree framework and adds the corner numbering the frontier engine uses. Five terms are used in sequence: the *scope* is the region of the X-Y space bounded by the theoretical or observed extremes of X and Y; the *expected empty space* is the part of that scope which should hold no observations if the hypothesised relation holds; the *boundary* is the theoretical line separating that space from the compatible region; the *frontier* is the boundary estimated from data; and *ceiling* and *floor* name the direction in which Y is bounded, a ceiling limiting how high Y can be for a given X and a floor how low.

Boundary type follows the empty corner and nothing else, so for sufficiency the boundary is a floor whenever the outcome level is high and a ceiling whenever it is low.

The direction is a claim about the theorised X-Y relationship, not about the shape of the estimated frontier, which may be a step function or a straight line under any of the four types.

Value

sca_corner() returns an integer vector; sca_direction() returns a character vector; sca_corner_map() and sca_terms() return data frames; sca_legacy_names() returns a named character vector whose names are the retired names and whose values are the replacements.

See Also

sca_scales, sca_table, sca_thresholds

Examples

```
sca_corner("HH")
sca_direction(4)
sca_corner_map()
```

```
sca_terms()
sca_legacy_names()
```

`sca_normalize` *Normalize Data and Diagnose Influential Observations*

Description

Provides min-max normalization and identifies observations or combinations of observations that materially influence the fitted empty-space effect size.

Usage

```
sca_normalize(data, scale = NULL, min_max = NULL)

sca_outliers(
  data, x, y, direction = "HH", ceiling = NULL, scope = NULL,
  k = 1, min.dif = 0.01, max.results = 25, condensed = FALSE
)

## S3 method for class 'sca_outliers'
print(x, ...)
```

Arguments

`data`	Input data frame or matrix.
`scale`	Output minimum-maximum pair, or one pair per column.
`min_max`	Optional source minimum-maximum pair, or one per column.
`x`	One condition column or an SCA outlier object in the print method.
`y`	One outcome column.
`direction`	One SCA direction.
`ceiling`	Frontier technique; CE-FDH is used when omitted.
`scope`	Optional theoretical scope.
`k`	Number of observations considered jointly.
`min.dif`	Minimum relative effect-size difference.
`max.results`	Maximum number of rows returned.
`condensed`	Suppress redundant combinations when `k > 1`.
`...`	Additional print arguments.

Value

`sca_normalize()` returns normalized data. `sca_outliers()` returns an outlier-sensitivity table or `NULL`.

Examples

```
dat <- data.frame(X = c(0, 2, 4), Y = c(10, 20, 30))
sca_normalize(dat)
```

`sca_plot` *Plot SCA Frontiers and Permutation Tests*

Description

Draws SCA-labelled scatter plots and fitted frontiers or the permutation distribution of an empty-space effect size.

Usage

```
sca_plot(model, x = NULL, ceilings = NULL, shade = TRUE, annotate = TRUE)

## S3 method for class 'sca_result'
plot(x, ...)

sca_test_plot(model, x = NULL, ceiling = "ce_fdh", bins = 30)
```

Arguments

`model`	An object returned by `sca_analysis()`.
`x`	Condition names or positions. In `plot.sca_result()`, the SCA result object.
`ceilings`	Optional frontier methods to draw.
`shade`	Shade the fitted empty zone when an exact CE-FDH or CE-VRS frontier is selected.
`annotate`	Label the logical direction and physical corner.
`ceiling`	Frontier method for the permutation test.
`bins`	Number of histogram bins.
`...`	Arguments passed to `sca_plot()`.

Value

A `ggplot` object for one condition, or a named list of plots for several conditions. The plot method returns the plot object invisibly.

sca_power *Power Analysis for Directional SCA*

Description

Estimates power through repeated samples and permutation tests, and plots the result.

A frontier line bounds only two of the four corners. A rising line bounds the upper-left and lower-right corners, so it suits "HH" and "LL"; a falling line bounds the upper-right and lower-left corners, so it suits "LH" and "HL". A horizontal line is accepted for every direction. sca_power() checks this before calling the engine, which would otherwise warn and return NULL.

Usage

```
sca_power(
  n = c(20, 50, 100), effect = 0.1, slope = 1,
  ceiling = "ce_fdh", direction = "HH", p = 0.05,
  distribution.x = "uniform", distribution.y = "uniform",
  rep = 100, test.rep = 200
)

sca_powerplot(
  df_power, x.variable = "n", x.name = "Sample size",
  x.min = 0, x.max = NULL, line.variable = "ES",
  line.name = "Effect size"
)
```

Arguments

n	Sample size or vector of sample sizes.
effect	Population effect size(s).
slope	Frontier slope(s). The sign must match the direction: zero or positive for "HH" and "LL", zero or negative for "LH" and "HL". Any other combination is an error.
ceiling	Frontier method(s).
direction	One SCA direction.
p	Significance level.
distribution.x, distribution.y	Marginal distribution(s).
rep	Number of datasets per design cell.
test.rep	Number of permutations per simulated analysis.
df_power	A data frame returned by sca_power().
x.variable	Column mapped to the x axis.
x.name	X-axis label.

x.min, x.max	X-axis limits.
line.variable	Column mapped to line colour.
line.name	Legend title.

Value

sca_power() returns a data frame; sca_powerplot() returns a ggplot object.

Examples

```
# Deliberately small: cost is length(n) x length(effect) x rep x test.rep
# engine fits, so a reporting-grade run uses rep = 100 and test.rep = 200
# and takes minutes rather than seconds.
power <- sca_power(n = c(30, 60), rep = 10, test.rep = 25)
sca_powerplot(power)
```

---

sca_random *Generate Data with a Directional Sufficiency Pattern*

Description

Generates observations in the unit square while excluding the physical corner implied by the requested sufficiency direction.

Usage

```
sca_random(
  n, intercepts, slopes, direction = "HH",
  distribution.x = "uniform", distribution.y = "uniform",
  mean.x = 0.5, mean.y = 0.5, sd.x = 0.2, sd.y = 0.2
)
```

Arguments

n	Number of observations.
intercepts	Frontier intercept or vector of intercepts.
slopes	Frontier slope or vector of slopes.
direction	One SCA direction.
distribution.x, distribution.y	"uniform" or "normal".
mean.x, mean.y	Means for truncated normal distributions.
sd.x, sd.y	Standard deviations for truncated normal distributions.

Value

A data frame with X condition columns and one Y outcome column.

Examples

```
set.seed(42)
dat <- sca_random(25, intercepts = 0, slopes = 1, direction = "HH")
head(dat)
```

`sca_reference` *Central-Tendency Reference Line*

Description

Fits the ordinary least-squares regression of the outcome on each condition, using the same observations the frontier was fitted to. This is the line `lm(y ~ x)` returns, and it is what `reference = "ols"` draws on `sca_plot`.

Usage

```
sca_reference(model, x = NULL)
```

Arguments

`model`	An object returned by `sca_analysis`.
`x`	Optional condition names or positions. All conditions by default.

Details

The line is reported separately from `sca_table` on purpose. Regression describes how the expected outcome moves with the condition; a sufficiency frontier describes which condition-outcome combinations are absent. Section 6.1 of the condition analysis in degree framework treats these as different targets, most informative when shown side by side rather than merged into one number, and section 4.3 notes that a frontier is fixed by the most extreme observations rather than by the central tendency. A reference line therefore has a slope and an intercept but no empty zone, no effect size, no permutation p value and no sufficiency threshold.

The line is computed on request, so it is available whether or not `reference = "ols"` was set. The `drawn` column records whether it was also passed to the engine and so appears on the plot.

Value

A data frame with one row per condition: `condition`, `outcome`, `line`, `line_type`, `observations`, `intercept`, `slope`, `r_squared` and `drawn`.

See Also

`sca_analysis`, `sca_table`, `sca_terms`

Examples

```
dat <- data.frame(X = seq(0, 1, length.out = 12),
                  Y = seq(0, 1, length.out = 12))
fit <- sca_analysis(dat, "X", "Y", direction = "HH",
                    ceilings = "ce_fdh", reference = "ols")
sca_reference(fit)
```

`sca_scales` *Reporting Scales for SCA Threshold Tables*

Description

`sca_scales()` lists the reporting scales understood by `sca_analysis` and `sca_thresholds`, with a note on how each behaves when the sufficiency direction refers to a low level. `sca_rescale()` applies one of those scales to arbitrary values, using exactly the conversion `sca_thresholds` uses internally.

Usage

```
sca_scales()

sca_rescale(
  values, reference, bounds = range(reference, na.rm = TRUE),
  scale = "percentage.range", letter = c("H", "L"),
  convention = c("absolute", "directional"),
  inequality = c("strict", "inclusive"), boundary = NULL
)
```

Arguments

`values`	Numeric vector in actual units.
`reference`	Observed values of the same variable, used by the `percentile` and `sd` scales.
`bounds`	Length-two axis bounds, including any theoretical scope. Defaults to the range of `reference`.
`scale`	One of the names returned by `sca_scales()`.
`letter`	`"H"` or `"L"`, the direction letter for this axis.
`convention`	`"absolute"` or `"directional"`.
`inequality`	`"strict"` or `"inclusive"`; affects the `percentile` scale only. Renamed from `boundary` in 0.4.0.
`boundary`	Deprecated. Former name of `inequality`.

Details

Five scales are available. `"actual"` keeps the original measurement units. `"percentage.range"` and `"percentage.max"` express a value as a percentage of the axis range or of the axis maximum. `"percentile"` reports the share of observations on the stated side of the value. `"sd"` reports standard deviations from the mean.

Under the `"absolute"` convention every scale is measured from the low end of the axis, whatever the direction, and the inequality carries the direction. Under the `"directional"` convention a low-level direction mirrors the axis, so the transformation decreases and the inequality flips. The `"actual"` scale is never mirrored, because reversing measurement units would produce values that do not occur in the data.

A percentile is a step function, so its strictness follows the inequality used by the rule. Pairing a strict < rule with $P(X \leq t)$ would drop the observation nearest the threshold once the data are expressed on the same scale.

`sca_rescale()` is exported because a threshold produced elsewhere is comparable with a sufficiency threshold only when both went through the same direction-aware mirroring and the same percentile strictness rule. The NSCA package uses it to place the necessity and sufficiency thresholds of one row on identical scales.

Value

`sca_scales()` returns a data frame with one row per scale, giving its description, whether it is mirrored for low-level directions, and a note on the mirrored form. `sca_rescale()` returns a numeric vector on the requested scale; non-finite inputs return `NA`.

See Also

`sca_analysis`, `sca_thresholds`, `sca_terms`

Examples

```
sca_scales()

x <- c(1, 3, 4, 7, 9)
sca_rescale(4, x, scale = "percentile", letter = "H")
sca_rescale(4, x, scale = "percentile", letter = "L")
```

`sca_table` *Summarize and Extract SCA Results*

Description

Creates tidy estimates, extracts a single parameter, translates fitted frontiers into directional sufficiency rules, or displays selected output.

Usage

```
sca_table(model, legacy = FALSE)

sca_extract(model, x = NULL, ceiling = NULL, param = "Effect size")

sca_thresholds(
  model, ceiling = "ce_fdh", x = NULL, scale = NULL,
  outcome_scale = NULL, convention = NULL,
  inequality = c("strict", "inclusive"), digits = 6L,
  legacy = FALSE, boundary = NULL
)

sca_output(
  model, summaries = TRUE, thresholds = FALSE, plots = TRUE,
  tests = FALSE, selection = NULL, ceiling = "ce_fdh",
  scale = NULL, outcome_scale = NULL, convention = NULL,
  inequality = c("strict", "inclusive")
)

## S3 method for class 'sca_result'
print(x, ...)

## S3 method for class 'sca_result'
summary(object, ...)

## S3 method for class 'summary_sca_result'
print(x, ...)
```

Arguments

model, object	An object returned by sca_analysis().
x	For extraction and thresholds, a condition name or position. For print methods, the object to print.
ceiling	A frontier method.
param	An NCA parameter name, an SCA metadata field ("direction", "sca_number", "empty_corner", "statement"), a geometry field ("relationship", "empty_space", "boundary_type"), or a name retired in 0.4.0.
scale, outcome_scale	Reporting scales for the condition and outcome axes. Both default to the scales requested in sca_analysis(). See sca_scales.
convention	"absolute" or "directional"; defaults to the convention requested in sca_analysis().
inequality	Use logically exact strict inequalities or an inclusive reporting convention for numeric frontier rules. Renamed from boundary in 0.4.0, which is reserved for the theoretical line separating an expected empty space from the compatible region.
boundary	Deprecated. Former name of inequality.

legacy	Append the column names retired in 0.4.0 as duplicates. See sca_legacy_names.
digits	Significant digits used in the formatted rule text.
summaries	Print the tidy summary.
thresholds	Print directional threshold rules.
plots	Draw frontier plots.
tests	Draw permutation-test plots.
selection	Optional condition names or positions.
...	Additional arguments reserved for methods.

Details

For an HH model, the default strict threshold rule has the form X > threshold => Y > outcome level. Strict inequalities follow exact contraposition at the fitted inequality. Set inequality = "inclusive" only when the measurement or calibration convention justifies including equality.

Thresholds are held in actual units in sufficiency_threshold_actual and outcome_level_actual, and converted to the reporting scale in sufficiency_threshold and outcome_level. A model can therefore be re-expressed on any scale without refitting. Under the "absolute" convention 0 sits at the low end of each axis and the inequality carries the direction; under "directional" the axis is mirrored for a low-level direction, which flips the inequality so that every rule reads as "more of the sufficient thing".

The status column states each row in sufficiency terms. This matters because the engine's own out-of-range markers invert under contraposition: a cell meaning "no minimum is required" for a necessity reading is exactly the case in which no attainable condition value reaches the outcome level.

estimable A threshold lies inside the scope.

no_threshold No attainable condition value guarantees this outcome level.

always_satisfied The threshold falls outside the scope on the permissive side, so the condition restricts nothing. This describes the condition side only: it does not assert that the outcome holds for every observation.

Three columns state the geometry in the vocabulary of condition analysis in degree. relationship gives the direction as a claim about the theorised X-Y relationship; empty_space names what the expected empty space would contain; boundary_type is "ceiling" when the boundary limits how high Y can be for a given X and "floor" when it limits how low. For sufficiency the boundary is a floor whenever the outcome level is high (HH, LH) and a ceiling whenever it is low (HL, LL), so the estimator names CE-FDH and CR-FDH keep the word "ceiling" even where the fitted boundary is a floor.

These are empirical frontier rules and should not be described as causal guarantees unless the design supports that interpretation.

Value

sca_table() and sca_thresholds() return data frames; sca_extract() returns one value; sca_output() invisibly returns the generated output; the methods return or print summary objects invisibly.

See Also

sca_legacy_names, sca_terms, sca_corner_map, sca_scales

Index